AD-A037 161 STANFORD RESEARCH INST MENLO PARK CALIF F/G 5/2
RESEARCH ON LARGE FILE MANAGEMENT INFORMATION SYSTEMS.(U)
JAN 77 M C PEASE, J GOLDBERG, D SAGALOWICZ N00014-71-C-0210
UNCLASSIFIED NL

1 OF 1
AD A037161

END

DATE FILMED
4-77

STANFORD RESEARCH INSTITUTE
Menlo Park, California 94025 · U.S.A.

(9) ANNUAL SUMMARY rept.,

(11) Jan[redacted]977

(12)

(6) RESEARCH ON LARGE FILE MANGEMENT INFORMATION SYSTEMS.

(10) By: Marshall C./Pease, III, Jack/Goldberg and Daniel/Sagalowicz

Prepared for:

(12) 24 p.

Office of Naval Research
Department of the Navy
Arlington, Virginia 22217

Contract Monitor: Marvin Denicoff
Program Director, Information Systems Branch

(15) Contract N00014-71-C-0210

332 500

SRI Project 1031

Approved by:

Jack Goldberg, Director
Information Science Laboratory

Earle D. Jones, Executive Director
Information Science and Engineering Division

332 500

mt

I INTRODUCTION

This is an annual summary of SRI's program of research on Large File Management Information Systems. In this report we will describe the present status of the research and our plans for the immediate future, our past accomplishments, and give a general description of ACS.1, an operationa, experimental system currently being developed.

II STATUS OF THE RESEARCH

An experimental interactive system called ACS.1, for Automated Command System, has been demonstrated. It has shown the feasibility of automatically constructing operational plans that satisfy a wide variety of constraints and requirements. Some constraints define the nature of the operation that is to be planned. These constraints are contained in what we call a "process model" of the operation. Other constraints limit the interactions of multiple operations that compete for resources. These constraints are contained in what we call the "resource models." The planning problem requires the construction of an operational plan that simultaneously satisfies specified objectives, and obeys the constraints imposed by the process and resource models, given the current and expected future state of the human organization. The capability of doing this has been demonstrated in ACS.1.

The ability of ACS.1 to assist in the administration of approved plans, and to monitor their execution, has also been demonstrated. The administrative function is exercised by ACS.1 issueing orders at the appropriate times for starting the various activities identified in the plan. The monitoring function uses its knowledge of when those activities should be completed if the plan is to be carried through successfully. If an activity that is part of a plan is not completed on schedule, ACS.1 recognizes the fact and seeks to replan the process. If it can do so, it advises the responsible human user of the fact, and continues with the revised plan. If it cannot, it issues an alert message, advising the responsible authorities of the need for command decision.

The file system that will support the retrospective analysis of operations has been developed but not yet integrated into ACS.1. A question-answering capability that would make the file system useful has not yet been developed, although it seems clear that it would be feasible for an important class of queries based on the process and resource models used by the system.

The user interface of ACS.1 has been implemented using a pseudo-natural language understander developed in the Artificial Intelligence Center at SRI. This provides a convenient way for the user to enter commands or queries to the system.

We have recognized the importance of making the system highly flexible so that it can be adapted to new operational conditions and requirements, and so that it can be implemented incrementally as experience with the system is gained. To this end, we have developed a design

for the schedulers, which are responsible for the assignment of resources, in which the resource model is clearly separated from the functions it uses and the data it holds. This development greatly enhances the user's facility to modify or adapt the rules and procedures of the schedulers according to his immediate needs and circumstance. The practicality of this design has been demonstrated in ACS.1.

We have also recognized the importance of providing a "context capability" so that data can be entered and manipulated on a contingent basis, either by the system itself, or by the user. The basic data structures and manipulative functions for this purpose are being introduced, and are expected to be fully operative before the end of this period. There will then remain the need for providing the functions that will support use of the capability in a way convenient to the user and to the system.

III PLANS FOR THE NEXT PERIOD (APRIL 1, 1977 THROUGH MARCH 31, 1978)

During the next period, we intend to revise the experimental system, ACS.1, in a number of significant ways. These include the following:

- Integration of the data base capabilities and its development as an active part of the system. The responsibility for monitoring the execution of approved plans, and for determining when replanning is needed, will be made a part of the data base subsystem. This subsystem will also exhibit at least a preliminary capability for data validation based on the knowledge held by the system in its various process and resource models. This will represent a significant advance in data base technology.

- The organization of the system as a whole will be revised to enhace its flexibility and to facilitate intervention as desired by the responsible human authority.

- The context capability being introduced will be developed to make it usable for developing and maintaining contingency plans and for testing the ability of the organization to respond to hypothetical situations.

- The design of the planners, which implement the process models, will be revised to facilitate adaptation or revision of their operation. This is expected to make it feasible to decompose a given process model into a hierarchical system of models, with resultant advantages in improving the flexibility and adaptability of the system.

The simulated operational environment of ACS.1 will be changed to a naval task force, with particular attention to its logistics. This change will be made for several reasons. First, the task force environment is critical to the Navy. Second, it will facilitate interaction with other programs being undertaken at SRI. Third, it will provide a new viewpoint from which to study the requirements that a command support system should meet. Finally, it will provide a test of the transportability to other application environments of the principles and techniques developed in ACS.1.

IV PAST ACCOMPLISHMENTS

A. Accomplishments Through 1975

The study is currently near the end of its sixth year. We will here review the accomplishments of the study through the end of 1975--i.e., until the start of the present period.

During the first year, the principal result was the development of the concept of a management support system. Key elements of the concept in its original form included the following:

- The separation of the management function from the administrative one, where the management function is defined as the identification of the need for policy and the setting of appropriate policy. The administrative function includes, then, the direction of operations according to established policy.

- The concept of utility files that contain a body of data sufficiently small to be handled in an interactive mode, but which may contain summary information derived from a possibly large base. The need for interactive capability was considered to be essential for the support of the managerial function.

- Adaptability throughout the system, but particularly within the utility file component, so that the system will remain responsive to the current managerial needs.

The results of the first year were reported in References 1-4.*

During the first year, also, visits were made to the Marine Corps Personnel Information System office, and to the office of the Navy Maintenance Support Organization, MSO, Mechanicsburg, PA, and Arlington, VA, to acquire some understanding of Naval systems and needs that use large files. As a result, a decision was made to use maintenance data as the vehicle for the research.

The principal results of the second year included:

- Implementation of a small interactive retrieval system based on a sample obtained from the Navy 3M (Material Maintenance Management) data base.

- Further development of the concept of adaptive utility files, including the study of certain techniques for the adaptive structuring of files to facilitate access.

---

*References are listed at the end of this summary.

- A preliminary study of the use of logical inference to facilitate dialogue between the user and the system, using the variables of the small sample data base of maintenance activities.

The results were reported in References 3-5, and presented in oral reports given at Mechanicsburg, PA and at ONR, Arlington, VA.

In addition, during the second year, one of the investigators attended a short course on the 3M system to obtain greater understanding of the nature and use of that data system.

In the third year, major developments included:

- Extension of the use of logical inference to provide a framework for obtaining goal-seeking behavior in the experimental data retrieval system using maintenance data.

- The introduction of QLISP, a language developed at SRI by the Artificial Intelligence Center for the implementation of goal-seeking behavior.

- The study of the requirements for an inferential alert capability that would be useful to managers.

- Investigation of the requirements for a high-level interaction language that would enhance the utility of the system for managers or their staffs.

- An analysis of various techniques for file compression.

As a result of this work, it was recognized that the system concept could be regarded as requiring a so-called knowledge-based design in which knowledge of some subject domain is incorporated as an integral part of the system and used to guide its response to queries or data.

The work of this year was reported in References 8-14, and presented in an oral report to the Fleet Material Support Organization (FSMO) at Mechanicsburg, PA.

During the fourth year, major results included:

- The development of a small experimental system for the implementation of alert functions using inductive inference on historical data.

- The development of the concept of what we call a "process model." A process model is analogous to a PERT chart, but without specification of the critical times or identification of specific resources. It describes generalized knowledge about a particular type of process. It reflects the way the manager views certain activities of his organization. It provides an

effective means for applying known constraints in generating management decisions. It also provides a useful structure for the corresponding data. It may be considered as defining a "frame" as the term is used in the artificial intelligence community, or as a type of "conceptual schema" as that term is used in data base theory.

These developments were reported in References 15-17. They were described at a planning meeting for a large-scale data bank conducted by NPRDC, and at a workshop held at SRI for representatives of the Fleet Materials Support Organization. Dr. D. Waltz, of the University of Illinois attended the latter.

During 1975, the principal accomplishments included:

- Development of the concept of a "process model" as a basis for the development of a system that could aid the manager in the planning of operations, the assignment of resources to planned operations, the administration and monitoring of planned operations, and the construction of historical files recording those operations.

- The development of the concept of a "resource model" that embodies the constraints that limit the utilization of a given type of resource.

- The design and implementation of an experimental system based on this concept was undertaken and carried to the point where it could be demonstrated early in 1976. For this system, we have adopted the name ACS.1,* for Automated Command System.

- The recognition of the principle that the rules and constraints embodied in the process and resource models should be separated from both the data and the data manipulating functions. This principle is recognized as of vital importance in making the system adaptable to the changing needs of the manager. The feasibility of such a separation has been demonstrated.

---

*The system was previously called SPADOR, for Scheduler, Planner and Administrator of Operations and Resources. The name has been changed since it has become clear that the principles implemented in it can be used in support of a wide range of command responsibilities. The addition of ".1" to the name is an acknowledgment that the present system is likely to be only the first of a series of systems incorporating increasing capabilities and addressing other command requirements.

This work has been reported in References 18-20. It was also reported as part of a paper entitled "Automatic Planning from a Frames Point of View," given by R. Fikes at a workshop on "Theoretical Issues in Natural Language Processing" held at Massachusetts Institute of Technology June 10-13, 1975. It was also discussed as part of a paper entitled "On the Representation, Generation, and Use of Plans: An Overview" by R. Fikes given at the 10th Annual Symposium at Carnegie-Mellon University, October 6-8, 1975.

The detailed structure of the experimental system, ACS.1, is discussed in the next section.

V DESCRIPTION OF ACS.1, AN EXPERIMENTAL SYSTEM

A. General Description

ACS.1, for Automated Command System, is the name we have given to an experimental system designed to test and demonstrate the feasibility of the concepts that have been developed. For demonstration purposes, it operates in the simulated environment of a Naval air squadron operating from a carrier. In this environment, it responds to orders from the commander directing that a specified mission be planned. Once the plan has been approved by the commander, ACS.1 assists in its execution, originating orders for the tasks necessary to its execution and monitoring that these tasks are completed as required. After completion of the mission, ACS.1 is intended to store the significant data about it in a way that will make the data accessible and useful for retrospective analysis.

The construction of a plan requires the instantiation of a process model. A process model is a generalized description of the process involved. It includes identification of the tasks that must be done as part of the process, such as the preparation of the aircraft, the pilot briefing, the launch, flight and recovery of the aircraft and pilot, and the post-flight operations. It identifies the module that knows about, and is responsible for, planning these tasks. It also includes the constraints on these tasks, such as that certain tasks must be completed first. The process model identifies the types of resources that are needed, the relations between the times of assignment of the resources and the tasks, and the identities of the modules that are responsible for the resource types. There can be a hierarchy of process models--the top level process model, for example, may identify the entire pre-flight preparation of the aircraft as a task, depending on another module to instantiate a process model that describes the tasks involved in preparing the aircraft, such as its fueling and arming.

A plan has been generated when the process model is fully instantiated--for example, when the pilot and aircraft have been committed, when completion and end times for all tasks in the process have been specified and when all tasks identified in the process model have been planned. The last requirement makes the definition of planning recursive.

To create a plan, specific resources must be assigned. In the case of a mission, the top level process model requires the assignment of a pilot and an aircraft. Subsidiary process models, such as that for the preparation of the aircraft, may require assignment of other resources, such as maintenance crews. These assignments must be made in a way that is consistent with other existing plans, and with expected events that affect the availability of resources. For example, if a pilot is expected to be on leave during some period, he will not be available for assignment unless that leave is cancelled. The rules and constraints that interrelate assignments and events for some particular type of resource, such as aircraft or pilots, are the resource model for that type. Included are policy requirements such as that specifying a rest period following a flight for a pilot, and the scheduled maintenance rules for aircraft. The resource model embodies the knowledge that must govern the utilization of the given type of resource.

The process models, then, describe the nature of the various processes and sub-processes. The resource models describe the interactions of the various plans, and the effect of other events. A plan is the instantiation of the appropriate process models that is consistent with other demands on the required resources, where consistency is defined by the corresponding resource model.

The design of ACS.1 implements the decomposition of the problem by process and resource models. The system configuration is shown in block form in Figure 1.

The user interacts with the system through a user interface module that provides a pseudo-natural language capability. That is, inputs, whether commands or requests for data, are in English format. However, the input must use one of a set of specified commands or questions with names, times, and other variables being expressed in a specified format. The interface is able to execute a pattern match on these inputs, and to construct the appropriate function call.

The function call generated by the User Interface is transmitted to the Message Handler which routes it to the appropriate module. The actual operations, whether planning or information entry or retrieval, are executed by a set of modules which we call "planners" and "schedulers." The planners are program modules each of which is able to obtain an instantiation of a process model, as is described shortly. The schedulers are program modules which hold and manipulate data about the various resource types, and which are controlled by the various resource models. All interactions among these modules are handled by messages that are transmitted through the message handler.

In addition, there is the data system that holds the data about executed plans and other events. This has not been fully implemented, as yet, but is included in the design concept. Access to it, whether by the user or from one of the planners or schedulers, is, again, through the message handler. As is discussed later, we currently see the need for other means of accessing the data system, but the capability of accessing it from a planner or scheduler through the Message Handler appears to be a necessary one.

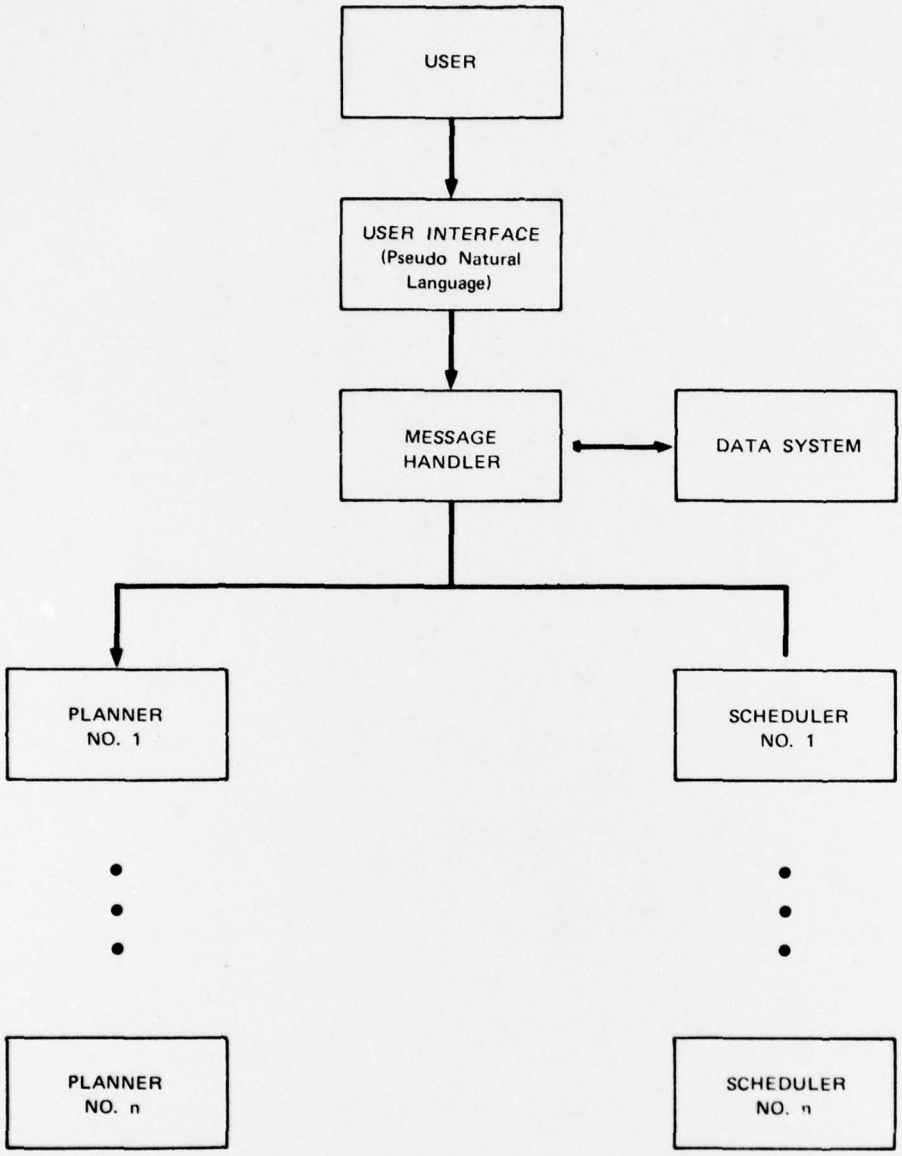

FIGURE 1 SIMPLIFIED BLOCK DIAGRAM OF ACS.1

Each planner contains within it a process model, which may be, for example, at the top level for missions, or at a lower level for such tasks as the preparation of an aircraft for flight. Each planner can accept a request, or order, to prepare a plan that will satisfy appropriate conditions--e.g., in the mission planner, that the flight shall reach a specified target area at a specified time, and shall depart that area at a given time. In response to such a request, the planner returns a plan only after it has been completely generated, including the complete planning of all of its tasks.

Each scheduler is responsible for some particular type of resource, such as aircraft, pilots, maintenance crews, and the launch facility. Each contains the names of the particular entities in the resource type for which it is responsible, and the corresponding resource model. A scheduler can accept requests for the assignment of one of its named resources. If so instructed or authorized, it makes the assignment itself if it can, committing that resource to activity being planned.

An inportant feature of the schedulers, the need for which was recognized in 1976 and which has been implemented, is the ability of the commander to specify what authority the scheduler should have, and to change it as the situation warrants. For example, he may want the pilot scheduler to make all assignments on its own authority most of the time, but want to reserve that authority to himself during critical periods, or for particular missions. When he is making the decisions himself, a request to the scheduler causes it to enter a dialogue mode in which it presents the relevant information to the commander and accepts his decision.

Another principle of the system is that, once a plan has been constructed and approved, each module, whether planner or scheduler, retains responsibility for its part of the plan. In effect, it has made a commitment to the source of the request, whether that source is the commander or another module, and has the responsibility to fulfill that commitment. For example, given an approved plan, the module responsible for aircraft preparation has the responsibility for completing the preparation of the aircraft by the planned time. The aircraft scheduler has the responsibility to furnish the specified aircraft over the specified period of time. The individual modules, then, monitor ongoing events to identify when the plan, or some part of it, requires replanning. For example, if an aircraft is found to require maintenance, this fact is reported to the aircraft scheduler. If that aircraft has been assigned to a planned mission, it is the responsibility of the scheduler to recognize the connection. Its response may be to initiate planning for the necessary maintenance if it can be completed before the start of the prior assignment. Or it may substitute another aircraft for the planned one, modifying the plan. Or, finally, it may cause a complete replanning of the mission. Similarly, the mission planner has a commitment to complete an approved plan for a mission. If it cannot maintain the plan as an executable one, it issues an alert message to the commander advising him of the difficulty and allowing him to take whatever action is appropriate.

The actual implementation of ACS.1, as it currently exists, is slightly different in that the monitoring operation that looks to make sure that the tasks of a process are completed on time is done by a special monitor module, rather than the separate planners. Also,

for reasons that are discussed later, we intend to shift this responsibility to the front end of the data base. The principle remains, however, that, if there is slippage in a plan, this fact is reported to the corresponding planner, and it is the responsibility of that planner to determine what should be done about it.

The mechanism that is used to develop a plan is one of negotiation between the appropriate modules. For example, suppose the mission planner issues a request to the aircraft preparation planner to plan the preparation of a given aircraft in a specified interval of time. The aircraft preparation planner will plan this task according to specification if it can. If it cannot, however, it will still return a plan that is as close as possible. In this case, the higher level planner will then determine if it can use the subplan despite the differences. If it cannot, it will, if possible, issue a new request with modified requirements that it can use, asking if these requirements can be met. Negotiation continues until either a plan is developed, or until it has been determined that no plan is possible.

If no satisfactory plan can be developed after all efforts at negotiation have been exhausted, this fact, together with the reasons for it, are referred back to the commander so that he can determine what to do about it. This is an instance of the principle that the system should not force solutions to exceptional, or non-routine problems. Such problems must be left to the decision of the commander.

Once a plan has been developed and accepted by the commander, the system aids in its administration and monitors its execution. For example, at the appropriate time, it issues orders that the pre-flight preparation of a specified aircraft is to be begun. It receives back data giving the status of the principal tasks at designated checkpoints--for example, at the completion of pre-flight preparation. Failing to receive such a notice, it initiates a query about its status and what can now be expected as its completion time. If the schedule slippage is too great--i.e., outside the tolerances that the planner has built into the plan--then the system initiates replanning. If it can replan around the new expected completion time of the task and still meet the requirements of the mission, it puts the revised plan into execution. If it cannot replan to meet the original conditions, it initiates an alert message advising the commander of the situation.

ACS.1, then, is able to plan activities that are described by process models, to maintain schedules for resources and to monitor their continuing availability for future plans, and to aid in the administration and monitoring of approved plans.

ACS.1 functions on a simulated clock basis. That is, it uses a simulated clock that is advanced only when all currently active processes have been completed, rather than operating on a real-time basis.

The organization of the planners and schedulers of ACS.1 parallels that of the human organization that exists to fill the same function. Therefore, the coordination of ACS.1 with the human organization can be adjusted to fit the immediate needs. In the context of a

naval air squadron, for example, if it is desired to maintain the pilot scheduling as a human function,, ACS.1 can be modified accordingly. In that case, the pilot scheduler would be bypassed and all messages addressed to it would be sent to a terminal. Replies to requests for assignment would be obtained from the terminal. The operations of the rest of the system would be unchanged.

ACS.1, therefore, could be brought on line incrementally. It could initially be introduced only for activities that are purely routine. Other activities that are not as clearly defined initially could be done in parallel by the human organization and by a module of ACS.1 until confidence is gained that all the necessary rules and constraints have been identified. ACS.1 would be permitted to assume the responsibility only when its ability to handle the task as required has been established.

Further, ACS.1 can be used flexibly. ACS.1 may have rules that will drive it appropriately under most circumstances, but not all. When exceptional circumstances are recognized, control can be switched back to the corresponding human organization. The determination of what activities are to be controlled by the system is itself controllable by the human authority.

The principle of making ACS.1 parallel the human organization is important for making it adaptable to the needs of the human organization and to the situation in which it operates.

B. System Design

An experimental system has been constructed and demonstrated on the PDP-10 of the Artificial Intelligence Center in SRI, using INTERLISP under TENEX. The principal features of this system include the following:

- The system is organized in a way that parallels the corresponding human orbanization. This has been recognized to be of great importance in facilitating the incremental introduction of such a system into an operational environment.

- The system exhibits a very high degree of modularity, as suggested by Figure 1. This is important since it facilitates system evolution by allowing the scope of the system to be extended as new modules are designed and their operation verified.

- A pseudo-natural language interface developed under another contract by the Artificial Intelligence Center in SRI has been incorporated into the User Interface of Figure 1. Its utility and convenience have been demonstrated.

- A file package has been designed for the data system of Figure 1, although it has not yet been integrated into the system. Integration has been deferred since the data system would be of little use without a question-answering front end. While such a front end appears

quite feasible, its design was judged to be less important for the research effort than the work on planning and on the maintenance of schedules.

- The capability of the system to monitor the execution of approved plans has been demonstrated. The implementation of this feature in the current design is expected to be changed. For reasons discussed later, it will be included in the front end of the data base. Nevertheless, the basic capability has been demonstrated.

In the following subsections, other features associated with specific functions and components of ACS.1, rather than with the system as a whole, are discussed.

C. Planner Design

The experimental system has been brought to the point where it demonstrates the feasibility of an automated planner for activities that are conveniently described by a hierarchy of process models. The principal structural features that have been implemented to achieve this result include the following:

- The development of a data structure that can encode a process model in a way that permits its instantiation into a plan. The information so encoded includes:

  1. The tasks that are included in the process.
  2. The constraints relating the tasks--e.g., that certain tasks must be completed before another task can be started, or that the order of certain tasks is immaterial, although they may not be concurrent.
  3. The assignments that must be made, and their relation to the tasks--e.g., that a resource of a given type must be assigned from the start of a given task to the end of another.
  4. The identity of the module that has responsibility for planning a task or making an assignment.
  5. The structure of the messages that will request planning tasks or assigning, and that will report completion of its planning activity.
  6. The expected duration of the tasks in the process, and the expected variation of the duration. (This is used in requesting assignments and in setting constraints on requests to plan tasks within the process.)

- The development of an internal priority system that is the basis of negotiation between modules. In requesting that a task be planned, an earliest start time (EST) and a latest end time (LET) are specified by the requester.

These are given priorities depending on whether they were derived from the initial requirement that the plan is to meet, from previous planning activities, or from the expected duration. If it proves impossible for the task planner to meet both the EST and the LET, it returns the best solution it can offer, as determined by the priorities. The requesting module can then either accept the proposed solution, or renew its request with higher priorities if this is permitted.

- In consequence of the above, the principle of negotiation among the modules has been established as leading to a workable procedure.

Techniques that implement these capabilities have been developed and tested in a simulated operational environment.

D. Scheduler Design

The schedulers play an important role in the system, providing the means through which the system keeps track of multiple demands for limited resources, and seeks to resolve conflicts between demands. We also see a potential value for the designs implementing the schedulers that is outside their role in planning. The schedulers of ACS.1 may be regarded as model-driven data systems, where the knowledge incorporated in the model is used to enforce the self-consistency of the data. Defined in this broad way, there appear to be a number of other types of applications that may be of interest, including such applications as the study of econometric or sociologic models and the analysis or control of power grids and other networks. While these possible applications are well outside the scope of this program, their recognition suggests that this aspect of the research program may have considerable significance.

As a preliminary remark, it is well to emphasize that the primary purpose of a scheduler is not to obtain an optimized schedule for the resources involved. The optimization of schedules is a problem in Operations Research rather than Computer Science, and, as such, has received much attention by other workers. Our goal is to provide a structure within which schedules and scheduling can be handled conveniently. Our initial objective has been to provide the means for maintaining a schedule, however derived, and for recognizing automatically when unexpected events have made an existing schedule unrealistic. When this happens, the scheduler is required to initiate the replanning or other activity that is appropriate to the new situation.

With this goal in mind, the functions that a scheduler must provide include:

- Accepting and recording all information that may affect the ability to execute existing plans, or modify future planning.

- Responding to requests for assignment of the scheduled resource. The source of the request may be a human or a planner. The required response may be the assignment of a named resource, or the initiation of interaction with a human decision-maker, as appropriate.

- Recognizing conflicts when they occur and taking the appropriate action. The required action may call for resolution of the conflict by rescheduling if possible, or by replanning if necessary. Or it may be to alert the cognizant human authority, advising him of the situations and possible actions, for his decision.

- Providing desired overviews to a human decision-maker of the expected status of the resource type. This is required even when the scheduler is making the assignments, since the human authority will need the capability of reviewing the expected status of key resources.

- Providing an automatic monitoring capability versus desired or undesired situations that will alert the responsible authority when a significant situation arises.

The demonstration system, as it currently exists, exhibits the desired capabilities. The techniques that have been found practical include the following:

- The "scroll table" (Ref. 20) as a means of retaining and displaying the required information. This has proven to be a technique that is useful, both for meeting planning requirements, and for responding to the manager's need for overviews of resource allocations.

- The use of demon functions* to achieve separation of data entry operations from side effects. This feature is important since it facilitates modification of the scheduler and the construction and introduction of new schedulers during system growth or evolution.

- The use of demon functions to provide evaluative alert capabilities to inform the manager of significant changes in the state of the resource type in general. A further use of this capability, which has not been implemented as

*A "demon function" is one that is set to watch certain data elements. When these data elements are changed by the entry of new data, the change is examined to determine if the preconditions specified by the demon are met. If so, the demon is "fired" and the demon function is executed.

To be complete, this describes "write demons." There can also be "read demons" that may be fired on reading data elements. However, we have had no occasion to use read demons.

yet, is its use to provide the system, rather than the manager, with evaluative measures. It appears that this capability can be used to improve the planning procedures. The use that has been made of the technique is to provide a flexible means for the manager to incorporate alert capabilities to meet his needs without requiring major reprogramming of scheduler functions.

. The explicit separation of the rules and constraints of the scheduler, the resource model, from the data-manipulating procedures or functions. The rules are entered in a special data structure that is associated with the form which makes them usable by all schedulers, being particularized to the conditions of the designated scheduler by reference to the special data structure for that scheduler. This technique makes it relatively easy to modify the operations of a given scheduler to incorporate new understanding or to meet special conditions. It also facilitates implementing new schedulers and so simplifies growth or evolution of the system.

With the exception noted under the evaluative use of demons, these techniques have been developed and tested in a simulated operational environment. They are effective and appear to be of great potential value to operational command.

E. Context Capability

We are currently introducing a "context capability." By this is meant the ability to introduce into the system data that may be tentative, or contingent on other factors.

A context capability is necessary to permit the system to hold contingency plans, so that the commitment of resources is made contingent on a later command decision.

It is also necessary to permit the user to explore the possibility or desirability of introducing certain changes. It should be noted that it is not sufficient for the user to make the changes, and then to undo them if they prove undesirable. There is too great a possibility that introduction of the changes may induce changes in other data as a side-effect. These side-effects would not be undone when the changes are undone. What is needed is a true rollback capability in which the original state of the scheduler is re-established. The easiest way to do this is to make the changes "in context," retaining the original state as a senior context. If the changes prove undesirable, the lower context is simply removed; if desirable, the lower context is promoted to become the senior one.

A context capability is also needed to improve the internal operation of the system. It appears necessary, for example, if the system is to make any attempt to optimize its schedules. All the optimal scheduling algorithms of which we are aware depend on first finding a feasible schedule--one which meets the stated requirements--and then exploring variations to determine if they are better according to some measure of quality. Such an algorithm requires that the current schedule

be held intact while a variation is developed "in context." The variation is then promoted to being the current version only if it is an improvement.

There is also considerable advantage to doing all planning "in context." It is assumed that no plan will be executed until approved by the responsible human authority. Hence, any plan that is constructed by the system should be regarded as tentative until approved, when it will be promoted to the top-level, or global, context. Further, it is necessary that the commander be able to explore possible modifications of an unapproved plan, without either losing the original plan, or forcing modification of other, approved plans. It seems very desirable, therefore, to hold all unapproved plans in an appropriate subordinate context.

We have begun the modification of the system to incorporate a context capability. By the end of the current period (March 31, 1977) we expect to have the basic capability fully implemented, although not fully integrated into system operation.

To achieve this capability, we are setting up a top-level module which we call the "context manager" (CTX-MGR). This is at the top level so that context designations will be consistent throughout the system. For example, all components of a tentative plan will be under a given context designator, regardless of what scheduler or planner contains them.

The context structure will be that of a tree, rooted in context #0, which is the global context. We have considered the possibility of using a lattice, rather than a tree, so that a given context could inherit information from more than a single parent context. We have been unable to find any way of assuring that consistency is maintained in a lattice type of structure. We will, therefore, use the simpler tree structure.

The implementation of the CTX-MGR is essentially completed. Work is progressing on revising the basic scroll-table functions to incorporate contexts.

The next step will be the implementation of the generalized scheduler functions in a context environment--the functions which enter and manipulate data in the schedulers after specialization to a particular scheduler through use of the resource model as it is encoded in the table header a-list.* This work, also, is expected to be essentially completed before the end of this period.

The final step will be the construction of the schedulers required for the chosen application environment. Since we propose changing the application environment, this will not be undertaken before the end of this period.

*The "table header" is a special data structure associated with the scroll table of a given scheduler which encodes the basic information about that table. One part of the table header is what is called an "a-list" for "association list." It permits retrieval of property values by the name of the property desired, and so is associative in character.

The modification of the planning capability to incorporate context manipulation appears simpler. It is anticipated that the only difference from the current design that is required is the retention of all components and values of the plan in the appropriate context. This will be undertaken before the end of this period if time permits.

VI PUBLICATIONS PRESENTATIONS DURING THE CURRENT PERIOD

On March 16 and 17, 1976, R. E. Fikes and M. C. Pease gave a review of the program including the results obtained with the demonstrable system and our plans, at a meeting called by ONR for an invited group of naval and Defense Department personnel.

On May 27, 1976, R. E. Fikes gave a talk to the Artificial Intelligence Seminar at Stanford University. The title of the talk was "SPADOR: A Scheduler, Planner and Administrator of Operations and Resources."

On July 9, 1976, R. E. Fikes gave a talk at Xerox Palo Alto Research Center entitled "SPADOR: A Scheduler, Planner and Administrator of Operations and Resources."

On October 22, 1976, R. E. Fikes gave a talk at the National ACM Conference in Houston, Texas, entitled "SPADOR: A Scheduler, Planner and Administrator of Operations and Resources."

On November 3, 1976, M. C. Pease gave a talk at the Joint National Meeting of the Operations Research Society of America (ORSA) and The Institute of Management Sciences (TIMS) at Miami Beach, Florida. The title of the talk was "Process Models in Inventory/Maintenance Systems."

A paper is currently in preparation. It describes the ACS.1 concept and the techniques used to obtain the features that are seen as important for management. The paper will be issued as a Technical Report prior to its submission to a technical journal.

REFERENCES

1. M. C. Pease and A. Waksman, "Large File Management Information Systems," Technical Report 1, Project 1031, Stanford Research Institute, Menlo Park, California (September 1971).

2. R. E. Bain, "A Bibliography for Management Support Systems," Technical Report 2, Project 1031, Stanford Research Institute, Menlo Park, California (December 1971).

3. M. C. Pease, "Large-File Management Information Systems," Proposal for Research, SRI No. ISU 71-131 (November 1971).

4. M. C. Pease, "Artificial Intelligence and Large File Management Programs at Stanford Research Institute," presented at a technical symposium on "Time Sharing and Multi-Access Computing in the 1970's" cosponsored by the Office of Naval Research and Systems Development Corporation (9-12 November 1971).

5. M. C. Pease and A. Waksman, "A Scenario for a Managerial Support Complex-Operational Standards," Technical Report 3, Project 1031, Stanford Research Institute, Menlo Park, California (May 1972).

6. J. Goldberg, M. C. Pease, and A. Waksman, "Progress Report on a Research Effort for the Office of Naval Research on Large File Management Information Systems," 3 October 1972. (Copies of viewgraphs of an oral presentation available from ONR.)

7. J. Goldberg, M. C. Pease, and A. Waksman, "A Large File Data Processing System," Proposal for Research, SRI No. ISU 73-9 (January 1973).

8. A. Waksman and M. W. Green, "On the Consecutive Retrieval Property in File Organization," IEEETC, Vol. C-23, No. 2 (February 1974).

9. A. Waksman, "On Generalized File Structure," Technical Report 4, Project 1031, Stanford Research Institute, Menlo Park, California (November 1973).

10. William H. Kautz, "File Compression for Simple Associative Search," Technical Report 5, Project 1031, Stanford Research Institute, Menlo Park, California (November 1973).

11. A. Waksman, M. C. Pease, and J. Goldberg, "A Program for a Management Support System," Technical Report 6, Project 1031, Stanford Research Institute, Menlo Park, California (November 1973).

12. M. C. Pease, "Self-Adaptive File Structures," Technical Report 8, Project 1031, Stanford Research Institute, Menlo Park, California (November 1973).

13. A. Waksman, "Information Retrieval and the Query Language," presented at the interface meeting by ACM in Interest Group on Programming Languages (SIGPLAN) and Interest Group on Information Retrieval (SIGIR), Gaithersburg, Maryland (4-6 November 1973).

14. A. Waksman, "The Interface Problem in Interactive Systems," Behavioral Research Methods and Instrumentation, Vol. 6, No. 2, pp. 235-236 (March 1974).

15. A. Waksman, "Extraction of Alert Functions by Inductive Inference on Historical Data," Technical Report 10, Project 1031, Stanford Research Institute, Menlo Park, California (October 1974).

16. M. C. Pease, "Application of a Process Model to a Management Support System," Technical Report 9, Project 1031, Stanford Research Institute, Menlo Park, California (July 1974).

17. J. Goldberg, R. E. Fikes, M. C. Pease, and A. Waksman, "Large File Management Information Systems," Proposal for Research, SRI No. ISU 74-199 (18 October 1974).

18. J. Goldberg, R. E. Fikes, and M. C. Pease, "Large File Management Information Systems," Proposal for Research, SRI No. ISU 75-205, (October 1975).

19. S. Weyl, "An Interlisp Relational Data System," Technical Report 11, Project 1031, Stanford Research Institute, Menlo Park, California (November 1975).

20. R. E. Fikes and M. C. Pease, "An Interactive Management Support System for Planning, Control, and Analysis," Technical Report 12, Project 1031, Stanford Research Institute, Menlo Park, California (November 1975).

21. Study Group on Data Base Management Systems, Interim Report, American National Standards Institute Document No. 7514TS01 ANSI, Washington, D. C., February 8, 1975.